Event Processing Architectures leading to an EPTS Reference Architecture

Adrian Paschke  Paul Vincent  Catherine Moxey  Themis Palpanas  Brian Connell  Alex Alves
Freie Universität  Tibco Software Inc.,  IBM, UK  University of Trento  WestGlobal  Oracle

ABSTRACT
We introduce a reference architecture for event processing, as defined by the EPTS reference architecture (RA) working group. An event processing reference architecture allows users to quickly create event processing solutions that adhere to known stakeholder requirements and architectural qualities, such as performance, scalability, and application coverage. The common EPTS reference architecture description is supported by the contributed event processing "architectures" from EPTS working group members, including multiple vendors and researchers.

Categories and Subject Descriptors
C.0; D.2.11

General Terms
System architectures; Domain-specific architectures

Keywords
Event Processing, Complex Event Processing, Reference Architecture, Event-Driven Architecture

1. INTRODUCTION
Event Processing (EP) is considered a new paradigm, or view, on information systems, and is realized in different vendor products and solutions. These incorporate various software technologies that provide an event-oriented view of systems, providing continuous and stateful views of incoming events. The use of multiple software techniques in high performance stateful event processing has resulted in associated specialized software and system architectures. To this end the Event Processing Technical Society (EPTS) set up a Reference Architecture group to provide a common Reference Architecture for event processing (EP). The goal is to define product- and system-independent abstractions and stakeholder views on current EP architectures and architectural practices.

2. TERMINOLOGY AND METHODOLOGY
The EPTS Reference Architecture description follows the ISO/IEC 42010:2007 standard methodology for architectural descriptions of software intensive systems. [1] This methodology defines an architectural description as a collection of documentations of the architecture addressing different views on the system for different stakeholders. The six important terminological elements are:

Architecture
The fundamental organization of a system embodied in its components, their relationships to each other, and to the environment, and the principles guiding its design and evolution.

Architectural Description
A collection of products that document the architecture.

System
A collection of components organized to accomplish a specific function or set of functions.

System Stakeholder
A system stakeholder is an individual, team, or organization (or classes thereof) with interests in, or concerns relative to, a system.

View
A representation of the whole system from the perspective of a related set of concerns.

Viewpoint
A specification of the conventions for constructing and using a view - a pattern or template which to develop individual views by establishing the purposes and audience for a view and the techniques for its creation and analysis.

For further information on the underlying conceptual reference model and reference methodology we refer to [2]. For the terminology of event processing concepts and components used in the EPTS RA description we also refer to the EPTS Glossary. [3]
For the EPTS Reference Architecture description we have currently defined two views – a functional view and a logical view – which address typical stakeholders such as the business analyst, the IT operations manager, the software architect / designer.

3. ARCHITECTURES
The EPTS RA group have combined "architectural diagrams" (representing product and system specific architecture information) from multiple sources - including vendors with customers using their technologies in everyday use - to derive some common architectural definitions. Some of these input architectures are based on implemented system architectures while others are a generic representation of possible specific architectures. At the time of writing two main architectural views have been described: logical and functional architectures.
4. Logical View

The Logical View (of the Event Processing Reference Architecture) (see figure 1) describes the logical layout of Event Processing Agents (EPAs) [3] in an Event Processing Network (EPN) [3].

![Logical View on EPTS Reference Architecture](image)

**Figure 1: Logical View on EPTS Reference Architecture**

It is important to note that the Logical View is an abstraction and not defined to some strict definition of EPA in terms of a concrete software technology which is used: the EPN represented in a logical view could itself form an "agent" in some higher level EPN. Conversely, an EPA in a Logical View could be defined as a subsidiary EPN. The Logical View is therefore an abstract view, and has the goal of describing the logical layout of EPAs relative to each other as well as to any specific event-handling network or distribution mechanism, from some perspective of interest (system or subsystem).

As a "view" of an EPN, the Logical View can itself be used in various contexts. Two typical uses or contexts are:

**Business Logical View** When a business analyst is viewing the overall event processing application, it is useful for them to understand the inputs, outputs and roles of the various EPAs. Therefore an "overview" of the application as a "business logical view" may be useful.

*Example: a business logical view may show the event flow from some sources through some EPAs and on through to management dashboards. This may be of interest because certain EPAs may be under the jurisdiction of different departments in the business.*

*Comparison: the business logical view may be compared to a high level process diagram or value chain, as a sequence of activities (in this case EPAs).*

**Operations Logical View** When an IT Operations Manager is viewing the overall EPN, it is useful for them to understand the numbers of operating EPAs and their associated performance characteristics. Therefore an "operational overview" of the system as an "operations logical view" may be useful.

*Example: an operations logical view may show active and inactive EPAs, and their associated operational Key Performance Indicators (KPIs), allowing operations staff to monitor the overall load on EPAs and thence deduce any operator actions (such as increase available resources for some EPA).*

*Comparison: the operations logical view is equivalent to systems monitoring dashboards and executive systems that are in widespread use in IT*

The logical view of an EPN and associated EPAs can also be considered as a model, or diagram, of an overall Event Driven Architecture (or EDA).

5. Functional View

The Functional View of the EP Reference Architecture focuses on the functions that can be required within an event processing system – the functions that can be spread across the EPAs in the EPN. The following explanation references the Functional View shown in figure 2.

![Functional View on EPTS Reference Architecture](image)

**Figure 2: Functional View on EPTS Reference Architecture**

In addition to run time functions (when events are produced, processed and consumed) there are associated design time functions (including such things as definitions of events and patterns of events to be detected), and also administration functions (including security administration and ensuring qualities of service for the system). In the figure, the design time and administration functions are shown to either side of the run time functions. This functional view is primarily focussed on automated event
processing operations, but it would also be possible for some or all of the functions to be manual operations.

5.1 Design Time

At design time, the events and event-based rules relevant to the particular event processing system are identified. Raw events, simple events, and derived events can be defined, along with the patterns of events that are significant to the system and rules which define how the derived (or complex) events arise from the emitted root events. Modeling is a design-time activity in which the expected or possible behaviours of events in the system, and how the events are to be handled, are modelled. The actions to be taken (event reactions) when events or patterns of events are observed are also defined. Design time functions also provide a means to query on the event definitions, patterns, regular expressions etc., and offer scope for improvements to the way the system responds to events.

5.2 Run Time

At the most basic level — as was shown in the Logical View — at run time, raw events are produced by an event producer or emitter, some event processing is carried out resulting in derived events which are ultimately consumed by an event consumer or sink. Events can be emitted into the system by event producers which might be devices such as sensors, monitors or probes, or elements of a system such as business processes, services, or applications. Events can be made available for event processing by such means as publication of events into a communication system, or retrieval of events from an event detection system. After processing, events are can be consumed by actuators, by dashboards which display events as they occur to some user community (ranging from IT system operators to business analysts), by business processes, applications or services which are driven as a result of the events, or other external reaction to the events produced by the event processing system. For the purposes of the reference architecture, event consumers are regarded as downstream consumers of events generated in event processing, but event processing can itself be regarded as an event consumer, and event consumers can in turn act as producers of events.

The layout of the run time portion of the figure shows that between event production and event consumption, a number of event processing functions might be carried out, depicted as Event Preparation, Event Analysis, Complex Event Detection and Event Reaction. However, none or all of these might be involved, some might be carried out more than once (represented by the "0.." cardinality on each), and the ordering of the functions is not mandated (although there is some degree of logical ordering). Within each of the functions there are multiple subfunctions, of which representative samples are shown.

- **Identification** – An important step to perform before the analysis of events is to detect situations where events reference the same named entities (e.g., products, locations, persons, etc.) in different ways. In order for the subsequent steps of event processing (i.e., filtering, enrichment, correlation, etc.) to be accurate, we need a mechanism that will tag all these references with the same identifier. The above functionality can be achieved through the use of the Entity Name System (ENS) [4] [5], which is a scalable infrastructure for assigning and managing unique, global identifiers for named entities.

- **Preparation** - Typically, the first processing is likely to be some form of selection from the events that have been received. The events may be Filtered on information in the event payload or in metadata, such that some subset is selected for further processing. Event adaption can convert events from an external format into some other format suitable for further processing, or for consumption. Events could also be Enriched by adding additional information to them from other data sources or from other events.

- **Analysis** - some form of computation might be carried out on events. Examples include: Identification of events, and potential removal of duplicates; Transformation of events, which normally acts on the event payload, and might convert the event to a different format, or normalize the event; Analytic techniques, which can include predictive capabilities such as trend computations; Tracking of events passing through the system, in terms of their location and time attributes; Scoring and Rating of events by computing values of events and their associated data, and Classification by identifying event types and associations.

- **Detection** - this processing covers aspects of deriving different ("complex" or derived) events from individual events, where multiple events might be Aggregated or Consolidated into a smaller number of events. Functions include Consolidation, in which additional event data can be added into complex events, Composition in which new events are created based on preceding events, and Aggregation, where information across multiple events is combined to provide summarized data. A key aspect to event detection is Pattern Matching, which enables new events to be derived by detecting a particular pattern of events, often across time, and event Correlation which allows related events to be correlated.

- **Reaction** - Event Reaction is identifying actions to be taken as a result of events, usually events arising from previous processing. This can include Assessment of a
change in situation that should be acted upon and Routing of events to appropriate destination(s). Event reaction can also involve advanced capabilities such as Prediction of future events or behaviours, using machine learning or predictive analytics, application of business rules to make decisions based on the events, and even the Discovery of new event types, event patterns and analyses.

Note that the box within which all these event processing functions appear does not represent any deployment configuration. For example, filtering of events down to a subset, or transformation of raw events, could occur at the event producer rather than being carried out within an event bus. It would also be possible for the processing, or aspects of it, to be federated across a number of event processing systems. Coming back round the figure to event consumption, it is possible for there to be further event selection on the complex events that have been detected. Furthermore, any of the functions could be implemented as internal components or as external services.

Administration

Administrative functions involve monitoring the correct behaviour of the event processing run time system, ensuring adequate performance and availability, controlling resource utilization for performance and other means, making updates to the system, and managing the security of the system. The security considerations can range from securing access to design-time definitions and models, controlling deployment of aspects of the run time function, and securing the production and consumption of events.

5. Conclusion and Future Work

The EPTS-RA WG plans to do further work to enhance and improve the different views of the Reference Architecture. The Reference Architecture will support future versions of the EPTS Glossary and assist with the application of EPTS Common Languages and other EP standards understanding.

6. About the EPTS Reference Architecture Working Group

The Event Processing Technical Society Reference Architecture Working Group started March, 2009. It currently has 18 members and is co-chaired by Adrian Paschke (RuleML) and Paul Vincent (TIBCO). Since July 09 it is merged with the EPTS Metamodel Working Group under a joint charter. Its scope is:

- Identify and utilize best practices and methods for technical architecture descriptions and interchange
- Liaise with relevant standards bodies for EP metamodels and reference architectures

7. ACKNOWLEDGMENTS

This work has been conducted within the scope of the Event Processing Technical Society Reference Architecture working group and the authors extend special thanks to the members David Tucker (Event Zero), John Morrell (Coral8, Inc.), Baden Hughes (Event Zero), Dieter Gawlick (Oracle), Alex Alves (Oracle), Anand Srinivasan (Oracle), Shailendra Mishra (Oracle), Zbigniew Jerzak (TU Dresden), Hans-Arno Jacobsen (University Toronto), Albert Mavashev (Nastel Technologies Inc.), Simon Courtenage (University of Westminster), Catherine Moxey (IBM), Richard Tibbetts (Streambase), Brian Connell (WestGlobal), Guy Sharon (IBM), Themis Palpanas (University of Trento).

8. References


[4] Heiko Stoermer, Themis Palpanas, George Giannakopoulos. The Entity Name System: Enabling the Web of Entities. International Workshop on Data Engineering meets the Semantic Web (DESWeb), in conjunction with the IEEE International Conference on Data Engineering (ICDE), Long Beach, CA, USA, March 2010